

Electronic device and method of enabling to animate an object

The invention relates to an electronic device, and in particular to an electronic device capable of determining a new animation for at least part of an interactive robot or interactive virtual character.

The invention further relates to a method of enabling to animate an object, and
5 in particular to a method of enabling to animate at least part of an interactive robot or interactive virtual character.

The invention also relates to a computer program product enabling upon its execution a programmable device to function as such an electronic device.

An embodiment of such an electronic device is known from "A User-Interface
10 Robot for Ambient Intelligent Environments", written by A.J.N. van Breemen, K. Crucq, B.J.A. Krose, M. Nuttin, J.M. Porta and E. Demeester, published in proceedings of ASER 2003, Bardolino, Italy, pp. 176-182. This article describes an interactive domestic robot with a 'real' face consisting of dynamic mouth, eyes and eyebrows. Each of these objects can have one of several positions. Animation of an object from one position to another position is
15 instantaneous. Although this allows the robot to quickly react to user input, it makes the robot's behaviour less believable and therefore communication between the robot and the user less efficient.

It is a first object of the invention to provide an electronic device of the kind
20 described in the opening paragraph, which enables relatively fluent animation of an object in an interactive environment.

It is a second object of the invention to provide a method of enabling to animate an object of the kind described in the opening paragraph, which enables relatively fluent animation of an object in an interactive environment.

25

The first object is according to the invention realized in that the electronic device comprises a processing unit capable of determining a first part of a new animation of an object on the basis of at least one position of the object in a first animation and on the basis of a first part of a second animation of the object and capable of determining a second

part of the new animation on the basis of a second part of the second animation. Instead of enabling animation of the first part of the second animation exactly as the author defined it, the electronic device advantageously enables instant reproduction of the second animation. As soon as user input is received that triggers the second animation, the first part of the new animation can be reproduced without causing an abrupt transition. The electronic device may be, for example, a consumer-electronics device in which a virtual character acts as a user interface for controlling the consumer-electronics device or it may be, for example, a robot.

The inventor has recognized that by applying audio-animatronics techniques to the known domestic robot, a believable interactive user-interface robot can be created.

Audio-animatronics – the technique of creating lifelike mechanical characters - is known from amusement parks. The mechanical characters are animated according to a pre-defined sequence of positions to create smooth lifelike movements. These audio-animatronics techniques can also be applied to other animations, for example to animations of virtual characters, e.g. animals or persons, used in computer games or used in other computer or consumer-electronics related applications.

The inventor has further recognized that simple strategies for applying audio-animatronics to the known method of animating an object are disadvantageous. If a new animation has to be performed in response to a stimulus, e.g. user input, while a first animation is being performed, a first simple strategy of waiting until the first animation ends in a neutral position before performing a second animation starting from the neutral position may lead to delays and therefore less-interactive behaviour. If the first animation does not end in the same position as the second animation begins, it may even be necessary to create an additional delay to create a smooth transition between the two animations, as described in US 2003/0191560. On the other hand, a second simple strategy of aborting a first animation in a first position, moving the (virtual or mechanical) object instantaneously to a start position of the second animation, and performing the second animation leads to the less-believable animations performed by the known domestic robot. In the present invention, a transition filter combines a part of the first animation (i.e. at least one position) and a part of the second animation during a transition period to create smooth transitions between animations.

The second object is according to the invention realized in that the method comprises the steps of enabling to animate the object during a first period on the basis of at least one position of the object in a first animation of the object and on the basis of a first part

of a second animation of the object and enabling to animate the object during a second period based on a second part of the second animation of the object. The first period is a transition period between the first animation and the second animation. In the second period, the displayed animation will generally be equal to the second part of the second animation.

5 A new animation S_i of an object i may be calculated by using equations (1) and (2) of Fig.7. In equations (1) and (2), t is the current time, t_t is the length of the first period (the transition period), t_1 is the start time of the first period and t_1+t_t is the end time of the first period and the start time of the second period. The first animation (for one object) is represented by the function S_i^A and the second animation (for the same object) is represented
10 by the function S_i^B . The second animation starts at time t_1 and ends after time t_1+t_t . The first animation starts before time t_1 . The first animation does not necessarily continue until time t_1+t_t : the first animation may be aborted at time t_1 or may end at a time t_2 between time t_1 and time t_1+t_t . In the first case, $S_i^A(t)$ is equal to $S_i^A(t_1)$ between t_1 and t_1+t_t . In the latter case, $S_i^A(t)$ is equal to $S_i^A(t_2)$ between t_2 and t_1+t_t .

15 In equation (2), the scalar α linearly depends on the time. Making it depend exponentially on the time will make the interpolation even smoother. In an alternative to equation (1), $S_i(t)$ may be written as a recursive function. Between t_1 and t_1+t_t , $S_i(t+\Delta)$ may, for example, be a linear combination of $S_i(t)$ and $S_i^B(t+\Delta)$.

20 The method of enabling to animate of an object may be performed, for example, by a manufacturer manufacturing an electronic device by the electronic device itself, by a software developer developing software involving a virtual character, by the software itself, and/or by a service provider running the software. The animation may be calculated and displayed on different devices. For example, a server on the Internet may calculate the animation and a client on the Internet may display the animation. The animated
25 object may be a whole robot or virtual character or a part (e.g. a mouth) of a robot or virtual character. An animation of a robot or virtual character may comprise multiple animations of parts of the robot or virtual character, each part having independent positions. In this case, it is advantageous to perform the method for each part independently, while using identical start and end times for the first period, i.e. the transition period.

30

These and other aspects of the method and electronic device of the invention will be further elucidated and described with reference to the drawings, in which:

Fig.1 shows a front view of an embodiment of the electronic device of the invention;

Fig.2 shows examples of facial expressions of the embodiment of Fig.1;

Fig.3 is a block diagram of the embodiment of Fig.1;

5 Fig.4 shows an animation of a facial expression of the embodiment of Fig.1;

Fig.5 is a block diagram showing details of two blocks of Fig.3;

Fig. 6 illustrates an animation of an object of the embodiment of Fig.1 performed with the method of the invention; and

Fig. 7 shows two equations used to calculate the animation of Fig.6.

10 Corresponding elements within the drawings are identified by the same reference numeral.

An embodiment of the electronic device is shown in Fig.1: an emotional user-
15 interface robot called iCat. iCat recognizes users, builds profiles of them and handles user requests. The profiles are used to personalize different kind of home automation functions. For instance, personalized light and sound conditions are used when a specific user asks iCat to create a 'relaxing atmosphere'. In order to learn rich user-profiles, a good social
20 relationship between the iCat and the user is required, because both should understand each other and be willing to spend time in teaching each other things about themselves. It is expected that a believable user-interface robot makes this relationship more enjoyable and effective.

Fig. 1 shows iCat's sensors and actuators. The robot is equipped with 13
standard R/C servos s1..s13 that control different parts of the face, such as the eye brows,
25 eyes, eye lids, mouth and head position. Fig. 2 shows some of the facial expressions that can be realized by this servo configuration. In the nose a camera cam1 is installed for face recognition and head tracking. iCat's foot contains two microphones mic1 and mic2 to record sound it hears and to determine the direction of the sound source. Also, a speaker sp1 is
30 installed to play sounds (WAV and MIDI files) and to generate speech. Furthermore, iCat is connected to a home network to control in-home devices (e.g. light, VCR, TV, radio) and to obtain information from the Internet. Finally, several touch sensors touch1..touch6 are installed to sense whether the user touches the robot.

User-interface robots should be both able to perform reasoning (e.g. about the user's profile and intentions) and to react fast to user input (e.g. when user touches the robot).

A *hybrid architecture* that offers deliberative as well as reactive capabilities fits these requirements best. Fig. 3 shows a common hybrid architecture. It consists of two layers that both receive sensor information and are able to access the actuators. The higher layer performs deliberative tasks such as planning, reasoning and task control. The lower layer performs behavior execution tasks. This layer contains a set of robot behaviors (control laws) that receive commands (e.g. setpoints, goals) from the higher deliberative layer. When a command is realized the robot behavior returns status information.

The field of Audio-Animatronics has developed engineering techniques to create lifelike characters. Their main approach is to build *prescribed* character performances, i.e. they program a script of servo, lights, sound and speech events that is being played when the character needs to perform. The advantage of this approach is that there is a precise control over the character's movements, which provides the opportunity to properly design them using principles of animation. This way, believable behavior is realized. The disadvantage is the lack of interactivity: the character cannot act in another way than its program prescribed. Fig. 4 shows an example of a pre-programmed animation script applied to the user-interface robot iCat. This script is used to let iCat fall asleep. Instead of just lowering the head and closing the eyes, animation principles are used to animate the iCat. First, *anticipation* is used to prepare the user that iCat is going to sleep. Letting iCat first yawn does this (the top five frames in Fig. 4). Secondly, the *slow-in slow-out* animation principle is applied. By making movements more slow at the extremes they become more natural. The end result is a robot that behaves apparent and understandable.

A *robot animation* is a sequence of actuator actions – e.g. servo, light, sound and speech actions – that animates the robot. The main issue in animating robots, i.e. in computing how the robot should act such that it is believable and interactive, is developing a computational model that calculates the sequences of device actions. Different categories of computational models can be distinguished:

Pre-programmed – The robot animation is stored in a table. Typically, such robot animations are hand-animated or generated from motion-captured data.

Simulated – The robot animation is defined by a simulation / mathematical model; e.g. a eye-blink model.

Imitation – The robot animation is learned online, e.g. while mimicking a human or other robot.

Robot behavior – A control law, which uses sensor signals to generate device actions, defines the robot animation; e.g. head tracking.

Instead of using one computational model to animate user-interface robots, it is more advantageous to use multiple models. Each model defines a separate robot animation that controls only a restricted set of the robot's actuators. This way, different computational models can be used: pre-programmed models for falling asleep and waking up, simulation models for eye-blinking and robot behaviors for camera-based head-tracking and lip-syncing when speaking. Unfortunately, using multiple models introduces several problems. First, the individual models need to be started and stopped at the right moment and under the right conditions. The deliberation layer of the hybrid robot architecture calculates these conditions. Another problem arises when executing multiple robot animation models. Individual animation events need to be synchronized, such that servo, light, sound and speech events happen at the same time instance. Also, the individual actions of simultaneously active robot animations need to be merged. Finally, unwanted transient behavior (e.g. abrupt changes) that arises due to the switching between robot animations need to be handled properly.

A robot animation engine was developed to handle multiple computational models for animating user-interface robots. This engine is part of the behavior execution layer in a hybrid robot architecture. While higher level deliberation processes generate commands to control robot animations, the engine itself deals with the specific merging problems described in the previous section. An abstract robot animation interface was used to integrate different computational robot animation models. This interface defines three elementary aspects of a robot animation. First, every robot animation has a unique *name* attribute. This name is used to refer to the particular robot animation. Secondly, a robot animation has an *initialize* method that is called each time the robot animation is (re-) started. During this call variables such as counters can be given an initial value. Lastly, a robot animation has a method to provide *the next animation event*. Every particular computational robot animation model is derived from the abstract robot animation interface. Each may have additional attributes and methods relevant for that computational model. For instance, a pre-programmed robot animation is loaded from disc and therefore has a special method for this. An imitation-based robot animation typically has a method to learn new animation events.

The robot execution engine is able to play several robot animations simultaneously, while handling the merging problems. Fig. 5 shows the architecture of the Robot Animation Engine and all its components:

Animation Library – Preloads and stores all robot animations.

Command Parser – Interprets commands received from a higher-level deliberation layer.

Animation Channel – Controls the execution of a single robot animation.

5 **Merging Logic** – Combines multiple animation events into a single event.

Transition Filter – Realizes a bumpless sequence of animation events.

Clock – Determines execution framerate of Animation Channels.

Animation Channels

10 Layering – the use of multiple animations – is a common technique to create and manage believable character behavior in games. The known concept of an *animation channel* is used to control the execution of multiple animations. In contrast to a robotic behavior-based architecture, animation channels can at runtime be loaded and unloaded with robot animations from the Animation Library. Different channel parameters can be set to
15 control the execution of the loaded robot animation. For instance, an animation channel could loop the animation, start with a delay, start at a particular frame or synchronize on another animation channel. Once the robot animation has been loaded and all parameters have been set, the animation can be started, stopped, paused or resumed.

20 **Merging Logic**

While prioritizing animations is a standard technique to merge animations, it is not able to handle all blending situations. Therefore a runtime configurable Merging Logic component is used, which provides the flexibility to use the animation engine for different situations, each requiring its own blending strategy. The specific blending configuration of
25 the Merging Logic component can be set at runtime on a per-actuator basis. For every individual servo, light, sound or speech channel a blending operator can be configured. The following blending operators are available:

Priority – Actuator actions with a lower priority are overruled by those with a higher priority.

30 (Weighted) Addition – Actuator actions are multiplied by a weighting factor and added.

Min / Max – The actuator action with the minimum / maximum value is selected.

Multiplication – All actuator actions are multiplied.

These operators are commonly used in both the area of robotics as well as animation. Additional known operators that could be added to extend the Merging Logic component include multiresolutional filtering, interpolation, timewarping, wave shaping and motion displacement mapping.

5

Transition Filter

Suddenly changing from one robot animation to another one may result into an abrupt transition. One technique to prevent this is using special key-frames to define start and end frames of robot animations. A new robot animation can only be started when its start
10 frame matches the end frame of the previous robot animation. This technique, however, cannot be applied to robot behaviors as the actuator actions are calculated at runtime from sensor inputs and internal variables. Therefore, a second technique is used: filtering. A Transition Filter component is used to realize smooth transitions between robot animations.

Fig. 6 illustrates the workings of the Transition Filter for a servo s_i . At time t_i
15 a switch occurs. During a limited time period, called the *transition period* t_i , the new servo animation s_i^B is combined with the last value of the previous servo animation s_i^A using the equations (1) and (2) of Fig.6. The Transition Filter calculates a linear combination of both robot animations during the transition period. The scalar α linearly depends on the time; making it depend exponentially on the time will make the interpolation even smoother.

20

Application

To evaluate the proposed Robot Animation Engine a scenario was developed in which the user-interface robot iCat manages lights and music in an Ambient Intelligence home environment called HomeLab. Speech was used to make requests to iCat. Besides
25 recognizing speech, iCat had to be able to perform head tracking, such that it keeps looking at the user while the user speaks, lip-syncing while it speaks to the user, eye-blinking to become more life-like and showing facial expressions to react properly to the users request (e.g. looking happy when the request was understood and looking sad when the request was unclear). Different computational models were used to realize these robot animations.

30

Five animation channels were defined to deal with the multiple robot animations. Table 1 shows these channels and describes their purpose. For instance, channel 0 is used for robot animations controlling all actuator devices (e.g. a falling asleep robot animation as shown in Fig. 4) and channel 2 is used by a lip-syncing robot animation to control the four servos of the mouth (s_8, s_9, s_{10}, s_{11} ; see Fig. 1).

Table 1

Channel	Name	Description
0	Full-Body	Plays robot animations controlling all devices (s1 ... s13, sp1).
1	Head	Plays robot animations controlling the head up/down (s12) and left/right (s13) servos, and the eyes (s5, s6, s7).
2	EyeLid	Plays robot animations controlling the eyelids servos (s3, s4).
3	Lips	To play robot animations controlling the four mouth servos (s8, s9, s10, s11).
4	Face	Facial expressions (s1 ... s13, sp1).

While the invention has been described in connection with preferred
embodiments, it will be understood that modifications thereof within the principles outlined
above will be evident to those skilled in the art, and thus the invention is not limited to the
preferred embodiments but is intended to encompass such modifications. The invention
resides in each and every novel characteristic feature and each and every combination of
characteristic features. Reference numerals in the claims do not limit their protective scope.
Use of the verb "to comprise" and its conjugations does not exclude the presence of elements
other than those stated in the claims. Use of the article "a" or "an" preceding an element does
not exclude the presence of a plurality of such elements.

'Means', as will be apparent to a person skilled in the art, are meant to include
any hardware (such as separate or integrated circuits or electronic elements) or software
(such as programs or parts of programs) which perform in operation or are designed to
perform a specified function, be it solely or in conjunction with other functions, be it in
isolation or in co-operation with other elements. The electronic device can be implemented
by means of hardware comprising several distinct elements, and by means of a suitably
programmed computer. 'Computer program' is to be understood to mean any software
product stored on a computer-readable medium, such as a floppy disk, downloadable via a
network, such as the Internet, or marketable in any other manner.